JAIF 2020
Journée thématique sur les attaques par injection de fautes

Discussion of the laser and EM Instruction Skip Fault Models

J.-M. Dutertre[1], A. Menu[1], T. Riom[1], O. Potin[1], J.-B. Rigaud[1], J.-L. Danger[2]

Jeudi 24 septembre 2020, ENS Paris

This research has been partially supported by the European Commission under H2020 SPARTA (Grant Agreement 830892)

(1) MINES Saint-Étienne
Une école de l'IMT

(2) TELECOM ParisTech
Une école de l'IMT

Institut Mines-Télécom

❑ Hardware attacks

- i.e. physical attacks targeting integrated circuits

- Fault injection attacks
  - Active perturbation attacks
  - Goal: induce an information leakage or gain unauthorized access
  - Considered injection means: laser illumination & EM perturbations
  - Target: microcontrollers
  - Fault model:
    - what an attacker can do,
    - instruction skip.

Outline

I. Laser/EM-induced instruction skip fault model

II. Experimental settings

III. Experimental results

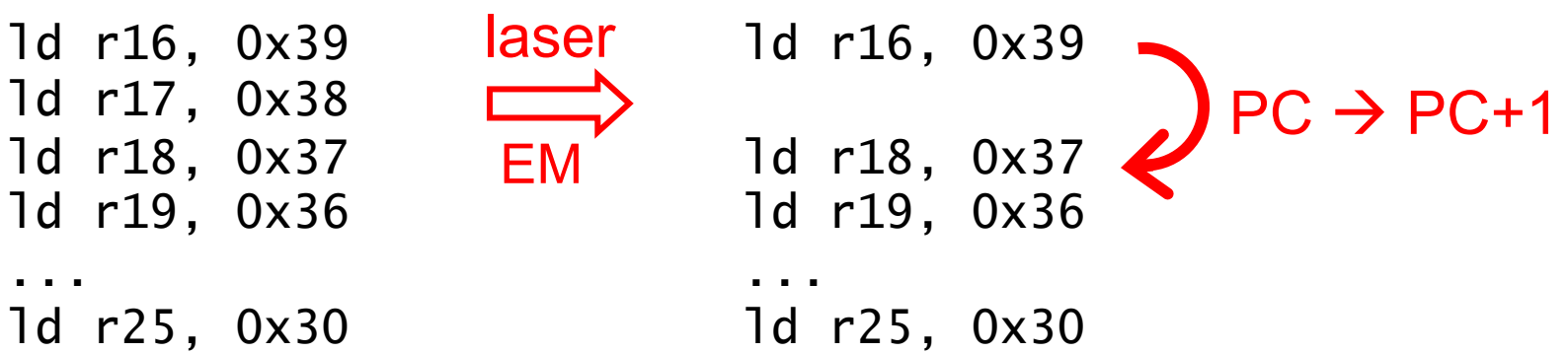
IV. Discussion

V. Conclusion

❑ Instruction skip fault model

Analysis of the laser/EM instruction skip fault model:

- Program Counter increase (PC → PC + 1)?

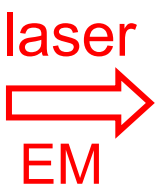


❑ Instruction skip fault model

Analysis of the laser/EM instruction skip fault model:

- Instruction modification (no operation, nop)?

```
ld r16, 0x39
ld r17, 0x38
ld r18, 0x37
ld r19, 0x36
...
ld r25, 0x30
```




```
ld r16, 0x39
nop
ld r18, 0x37
ld r19, 0x36
...
ld r25, 0x30
```

❑ State of the art

- EM-induced instruction skips
  - Single instruction skip:
    - 8-bit uCTRL [BEC19] (same target)
    - 32-bit uCTRL [MOR14]
  - Several instruction skips:
    - replay of 4 instr. at readout from cache [RIV15]
    - pipeline's instruction [YUC16]

⇨ Side effect of the target's microarchitecture

[BEC19] Beckers et al., Characterization of EM faults on ATmega328P, IEEE SEC 2019
[MOR14] Moro et al., Experimental evaluation of two software countermeasures against fault attacks, HOST 2014
[RIV15] Rivière et al., High precision fault attacks on the instruction cache of ARMv7-M architectures, HOST 2015
[YUC16] Yuce et al., Software fault resistance is futile: Effective single-glitch attacks, FDTC 2016

❑ State of the art

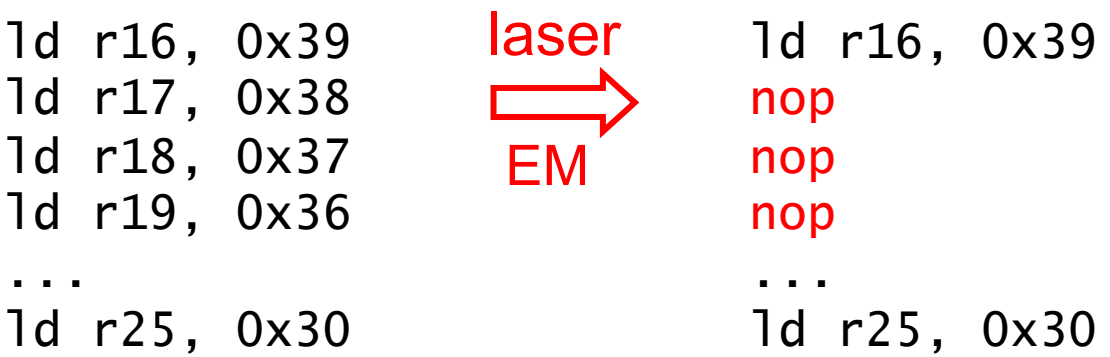
- Laser-induced instruction skips
  - Single instruction skip:
    - 8-bit uCTRL [BRE15, KUM18] (same target)
    - 32-bit uCTRL, CRT-RSA [TRI10], 2 skips distant from 58 ms

⇨ State of the art = single instruction skip

[BRE15] Breier et al., Testing feasibility of back-side laser fault injection on a microcontroller, WESS'15
[BRE15] Breier et al., Laser profiling for the back-side fault attacks: With a practical laser skip instruction attack on AES, CPSS'15
[KUM18] Kumar et al. An in-depth and black-box characterization of the effects of laser pulses on ATmega328P, CARDIS 2018
[TRI10] Trichina et al., Multi fault laser attacks on protected CRT-RSA, FDTC 2010

❑ Research objective

- Laser/EM-induced single instruction skip
  - accuracy, success rate, etc.
- Ability to extend the fault model (w.r.t. countermeasure design)?



Single nop

Several consecutive nops

❑ Research objective

- Laser/EM-induced single instruction skip
  - accuracy, success rate, etc.
- Ability to extend the fault model (w.r.t. countermeasure design)?

```
ld r16, 0x39
ld r17, 0x38
ld r18, 0x37
ld r19, 0x36
...
ld r25, 0x30
```

laser / EM ⇨

```
ld r16, 0x39
nop
ld r18, 0x37
nop
...
ld r25, 0x30
```

Single nop

Several consecutive nops

Several non-consecutive nops

Outline

I. Laser-induced instruction skip fault model

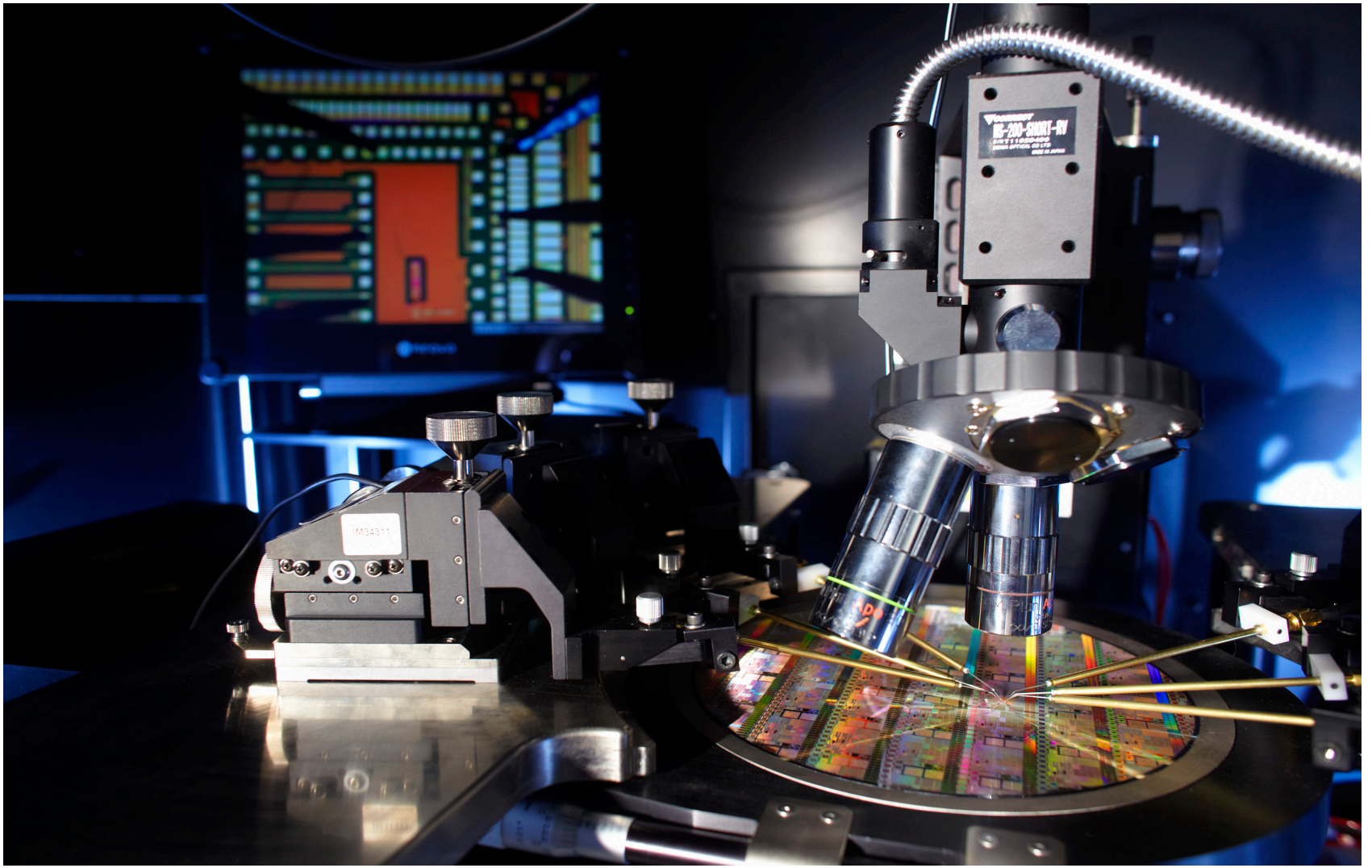
II. Experimental settings

III. Experimental results

IV. Discussion

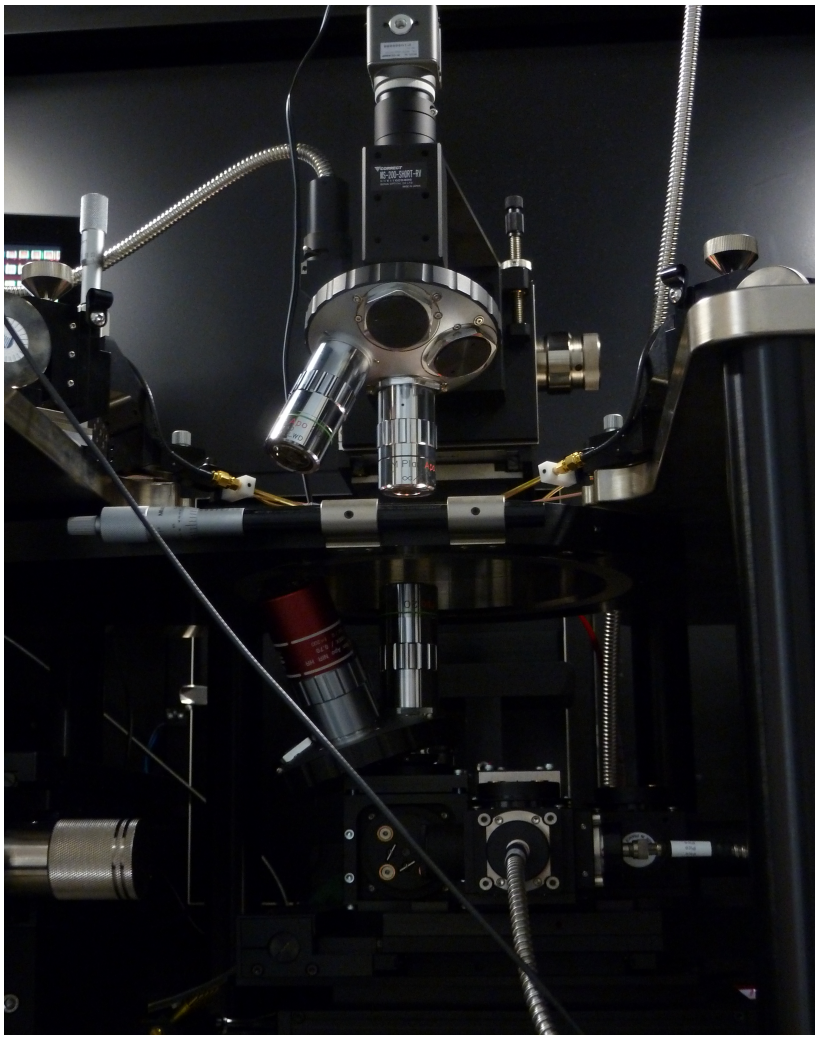
V. Conclusion

❑ Laser bench



❑ Laser bench

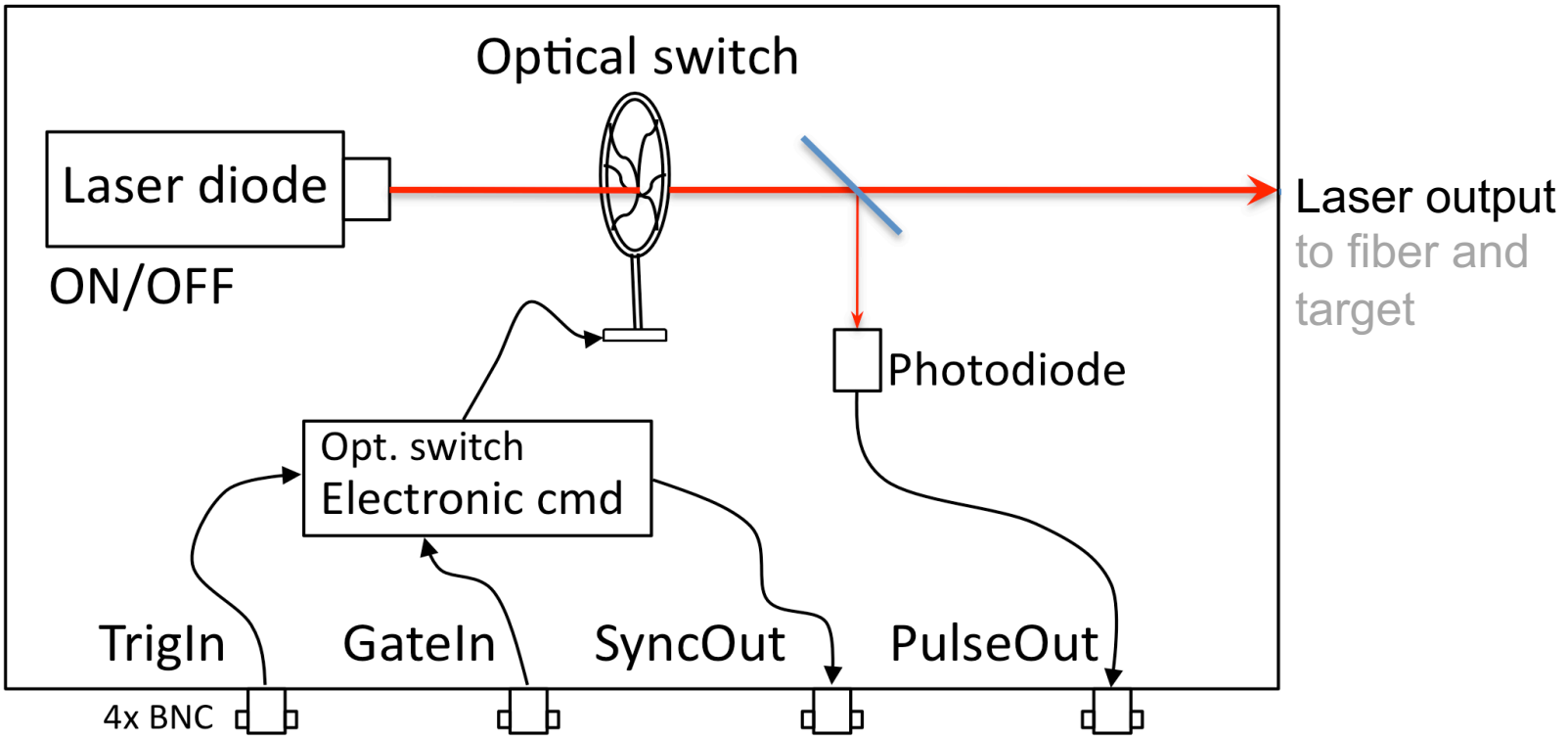
▪ Laser settings



- Rear side injection
- Wavelength: 1,064 nm (NIR)
- Spot size: 5 µm
- Pulse width: 50 ns – 1 s
- Energy max: 3 W
- Jitter: < 1ns

- IR camera
- Laser output (photodiode)
- XYZ stages : 0.1µm resolution

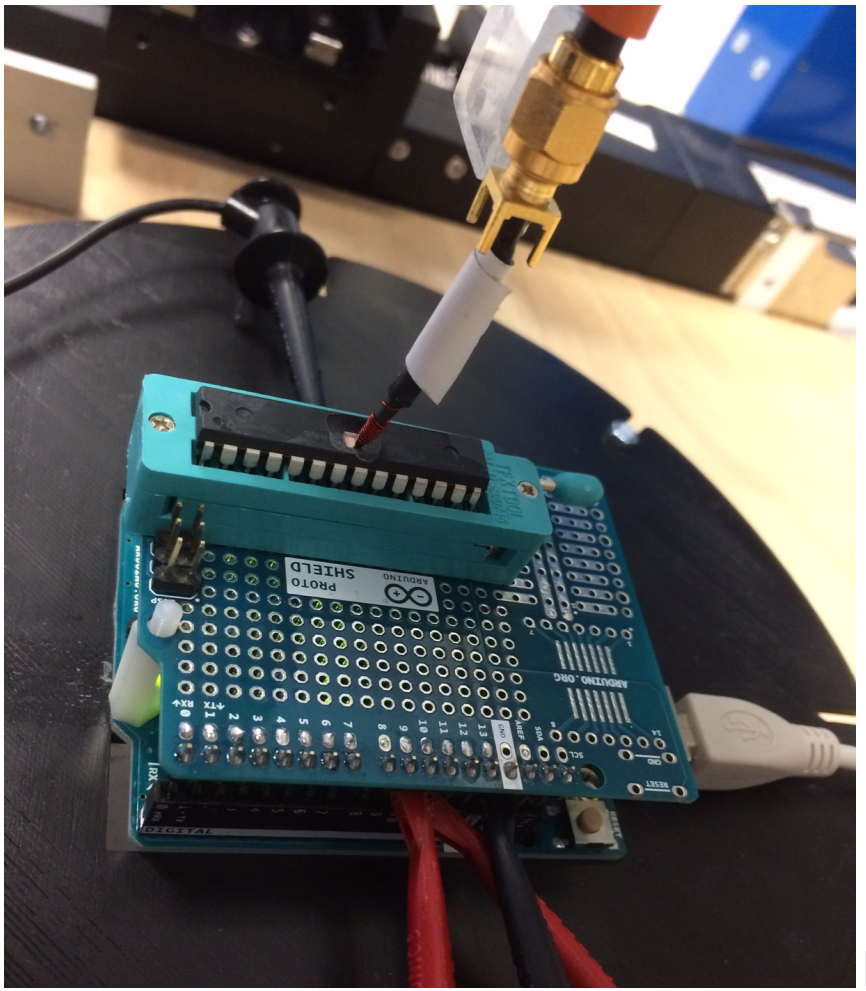
❑ Laser technology



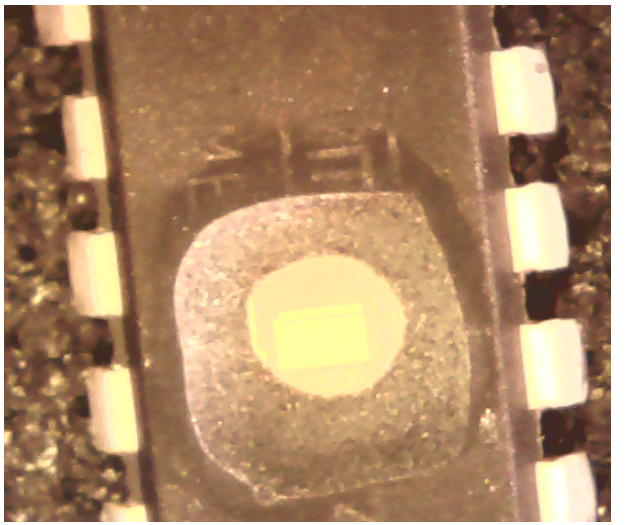
- Laser diode: ON/OFF (1,064 nm, IR)
- Optical switch & command: laser pulse shaping (25 ns opening/closing time)
- Beam splitter & photodiode: image of the laser pulse

❑ EM injection bench

- The EM injection bench



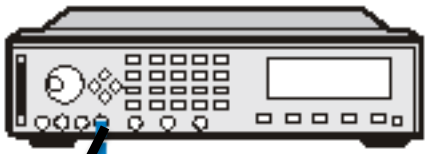
EM injection bench



Target close up

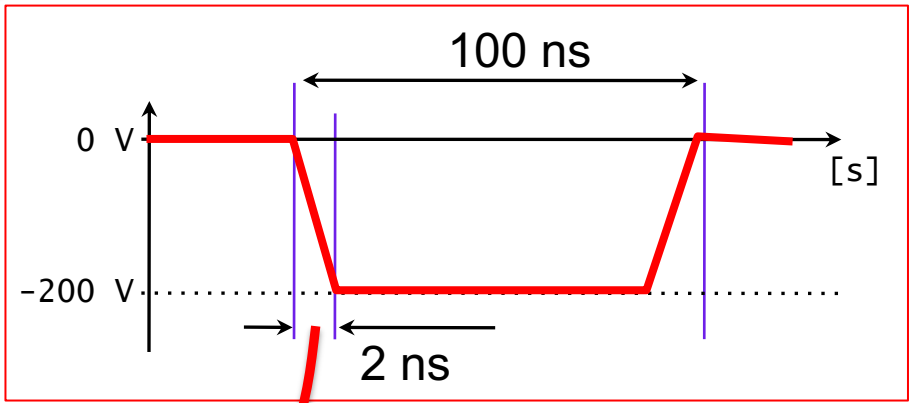
- The EM injection bench

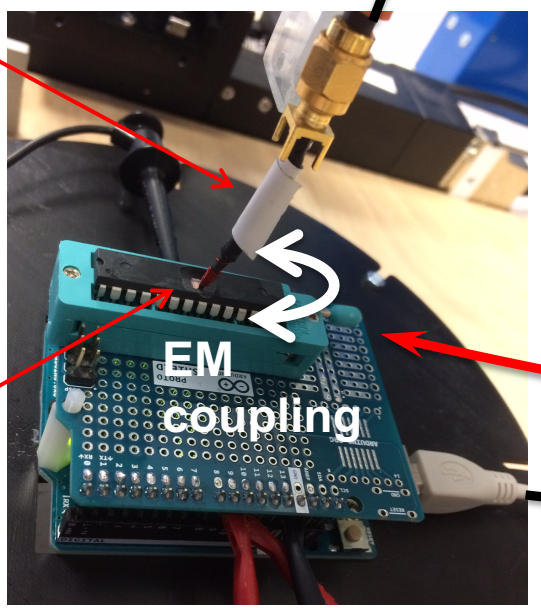
EM injection probe

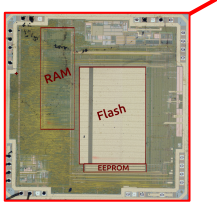


Pulse generator

Voltage pulse







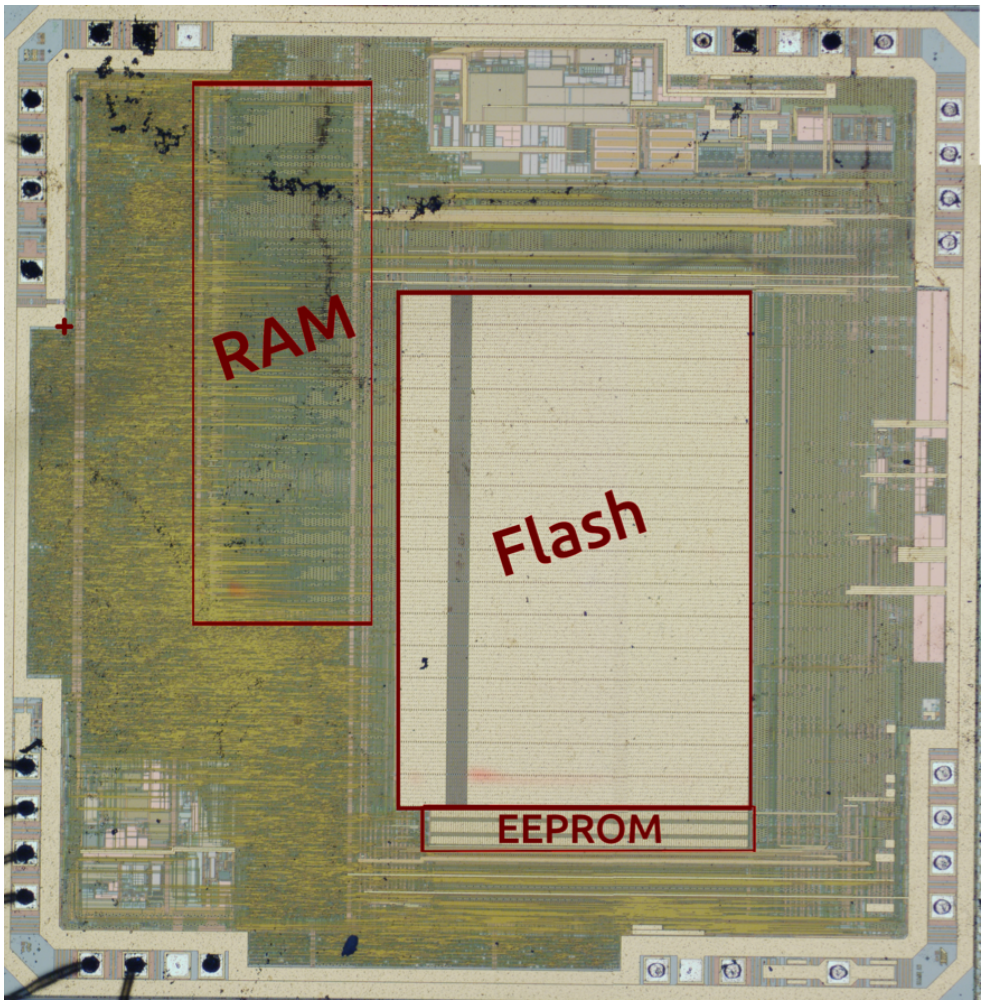
The target

USB serial

> ruby –I ./lib
> python …

+ EM injection probe: mounted on XYZ displacement stages (µm accuracy)

- The target: as simple as possible
  - 8-bit microcontroller: ATmega328P, 16 MHz (no pipeline)



RAM 2kB
Flash 3kB
EEPROM 1kB
32 gp registers
Fetch/execute

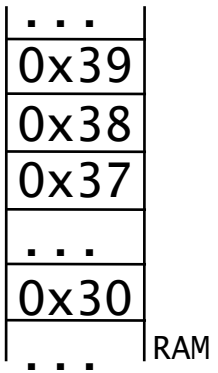
❑ The test code

- Goal: instruction skip analysis

Initialization

```
ldi r16, 0x55
ldi r17, 0x55
ldi r18, 0x55
...
ldi r25, 0x55
```

Z ⟶



Test code part

```
ld r16, Z+
ld r17, Z+
ld r18, Z+
ld r19, Z+
ld r20, Z+
...
ld r25, Z+
```

Register	16	17	18	19	20	21	22	23	24	25
Expected	0x39	0x38	0x37	0x36	0x35	0x34	0x33	0x32	0x31	0x30

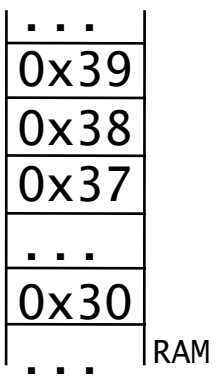
❑ The test code

- Goal: instruction skip analysis, instruction skip's effect

Initialization

```
ldi r16, 0x55
ldi r17, 0x55
ldi r18, 0x55
...
ldi r25, 0x55
```

Z ⟶



Test code part

```
ld r16, Z+
ld r17, Z+
ld r18, Z+
skip
ld r20, Z+
...
ld r25, Z+
```

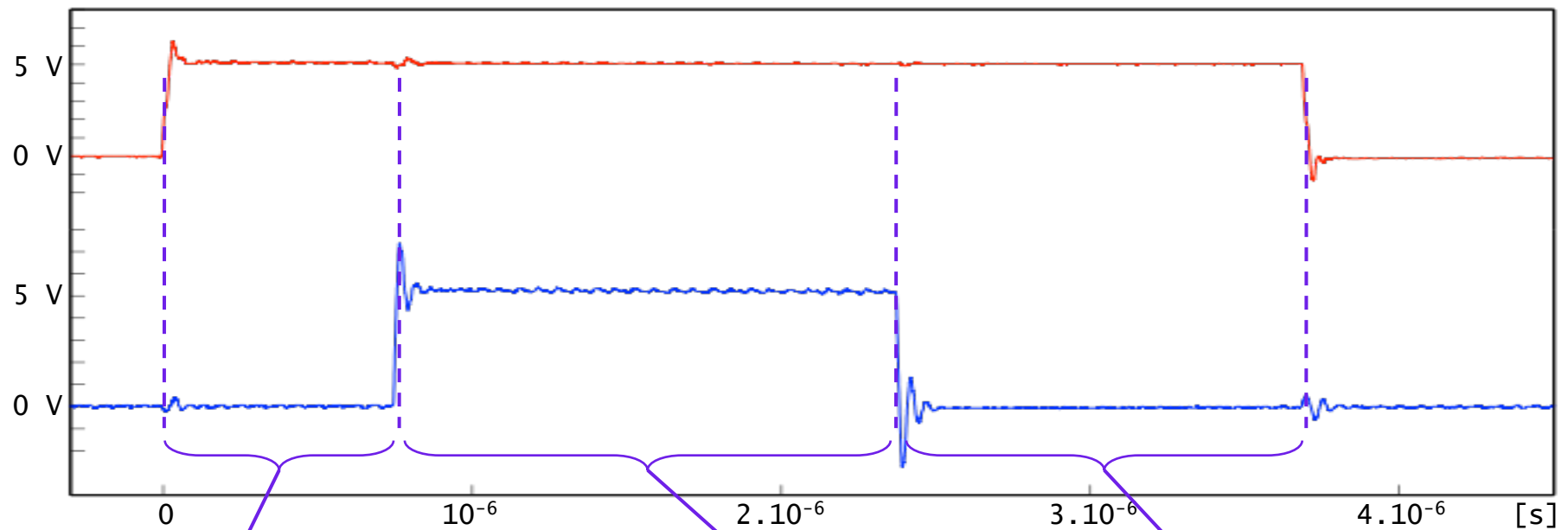
Register	16	17	18	19	20	21	22	23	24	25
Expected	0x39	0x38	0x37	0x36	0x35	0x34	0x33	0x32	0x31	0x30
Read	0x39	0x38	0x37	0x55	0x36	0x35	0x34	0x33	0x32	0x31

shift

(increment of adress Z was skiped)

❑ The test code

- Synchronization: Synchronization and core triggers



Initialization

```
ldi r16, 0x55
ldi r17, 0x55
ldi r18, 0x55
...
ldi r25, 0x55
```

Z → RAM: ... | 0x39 | 0x38 | 0x37 | ... | 0x30 | ...

Test code part

```
ld r16, Z+
ld r17, Z+
ld r18, Z+
ld r19, Z+
ld r20, Z+
...
ld r25, Z+
```

Registers readback

Outline

I. Laser-induced instruction skip fault model

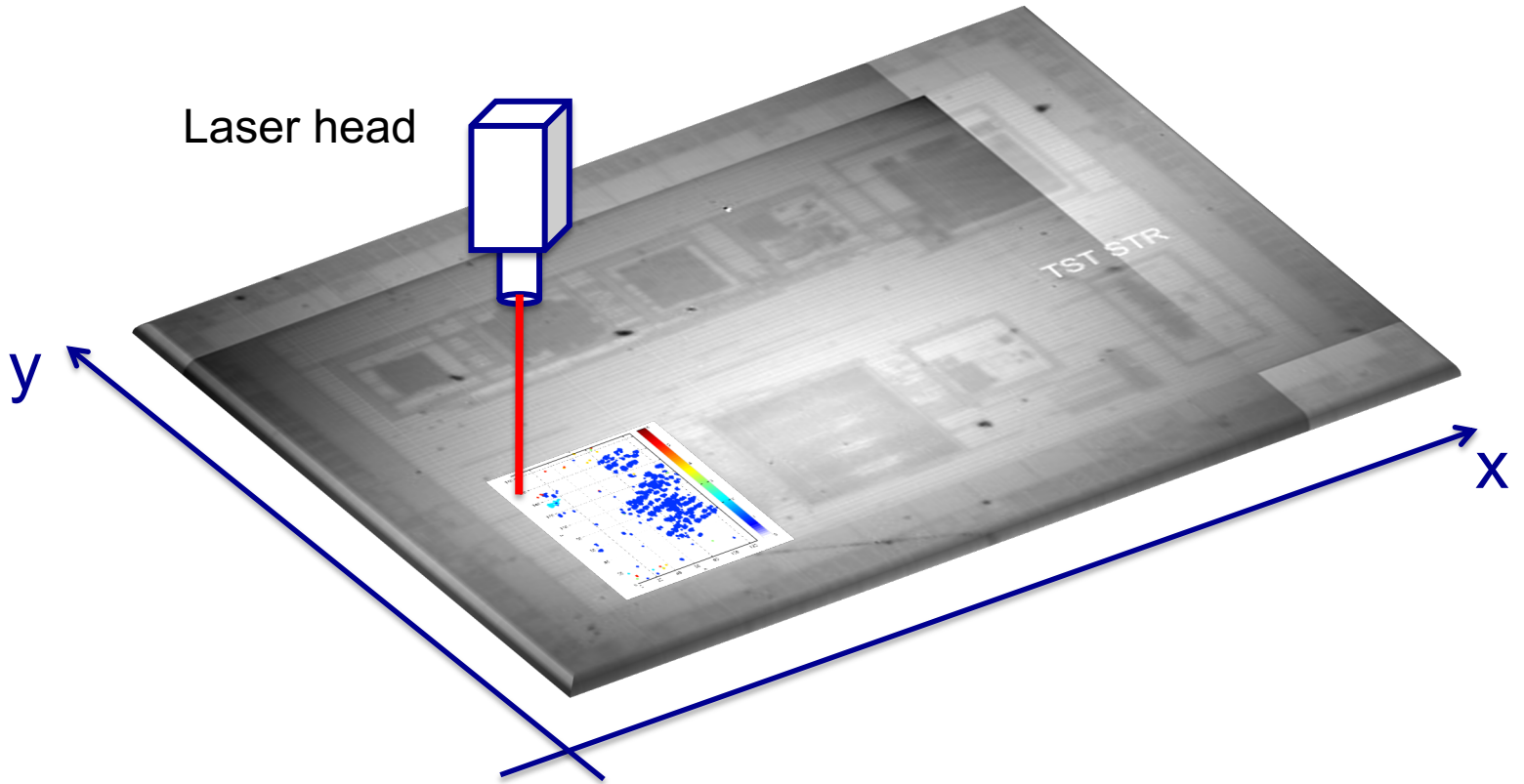
II. Experimental settings

III. Experimental results

IV. Discussion

V. Conclusion

❑ Experiments description



⇨ Laser fault sensitivity maps drawing
(colors according to the fault model)

❑ Finding the Points-of-Interest

Laser settings: 200 ns (3x Tclk), 0.5 W
XY steps: 50 µm
Synchronization: `ld r19, Z+`

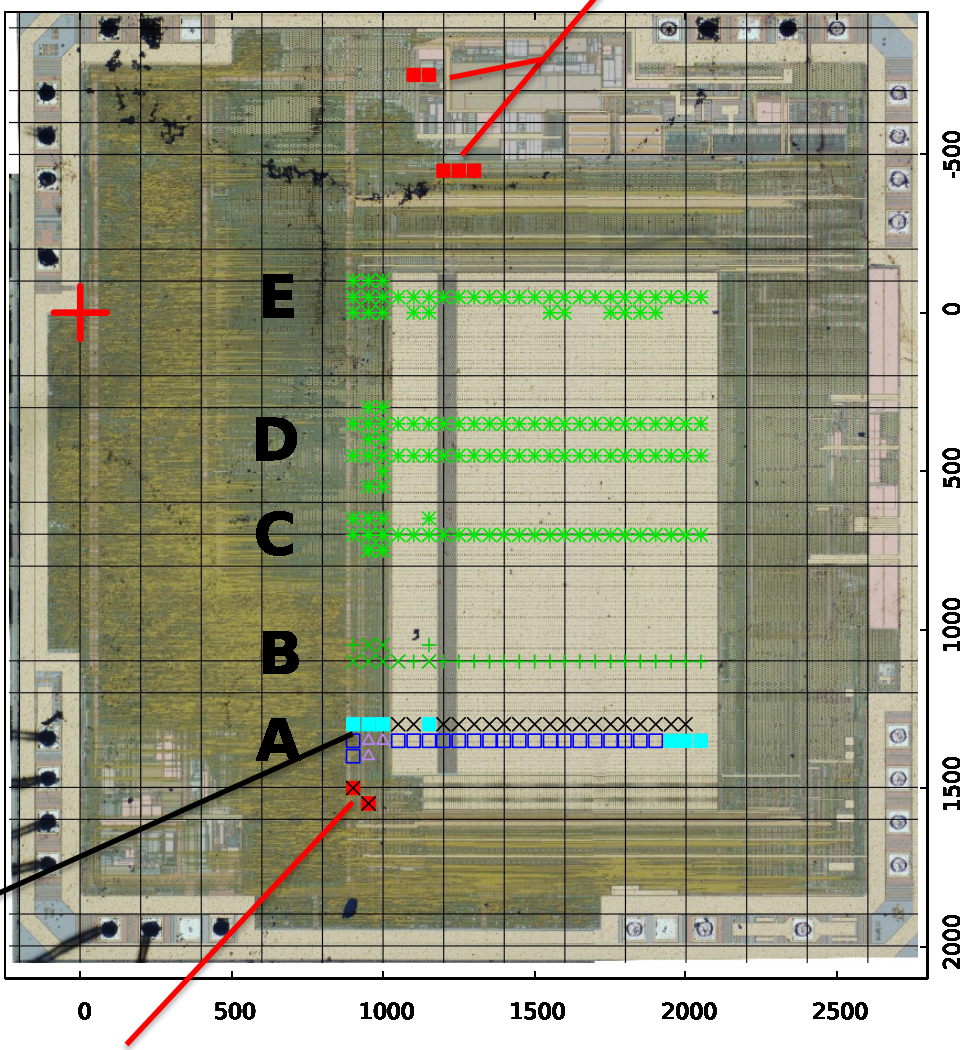
Fault key

S0	S1	S2	S3	S4	S5	S6	S7	S8	S9	SA
E0	E1	E2	E3	E4	E5	E6	E7	E8	E9	EA

Sx: color code
initialization value readback x times

Ey: shape code
other error values readback y times
(shifted values)

S2E5: 2x 0x55 + 5 shifted values
S3E4: 3x 0x55 + 4 shifted values



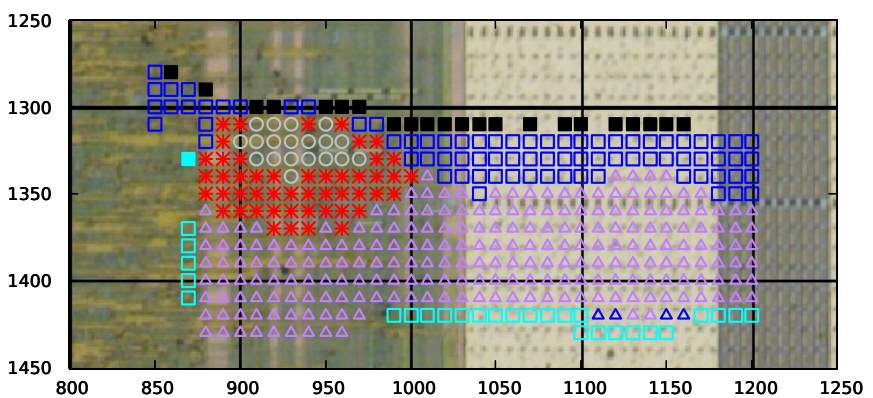
❑ Finding the Points-of-Interest

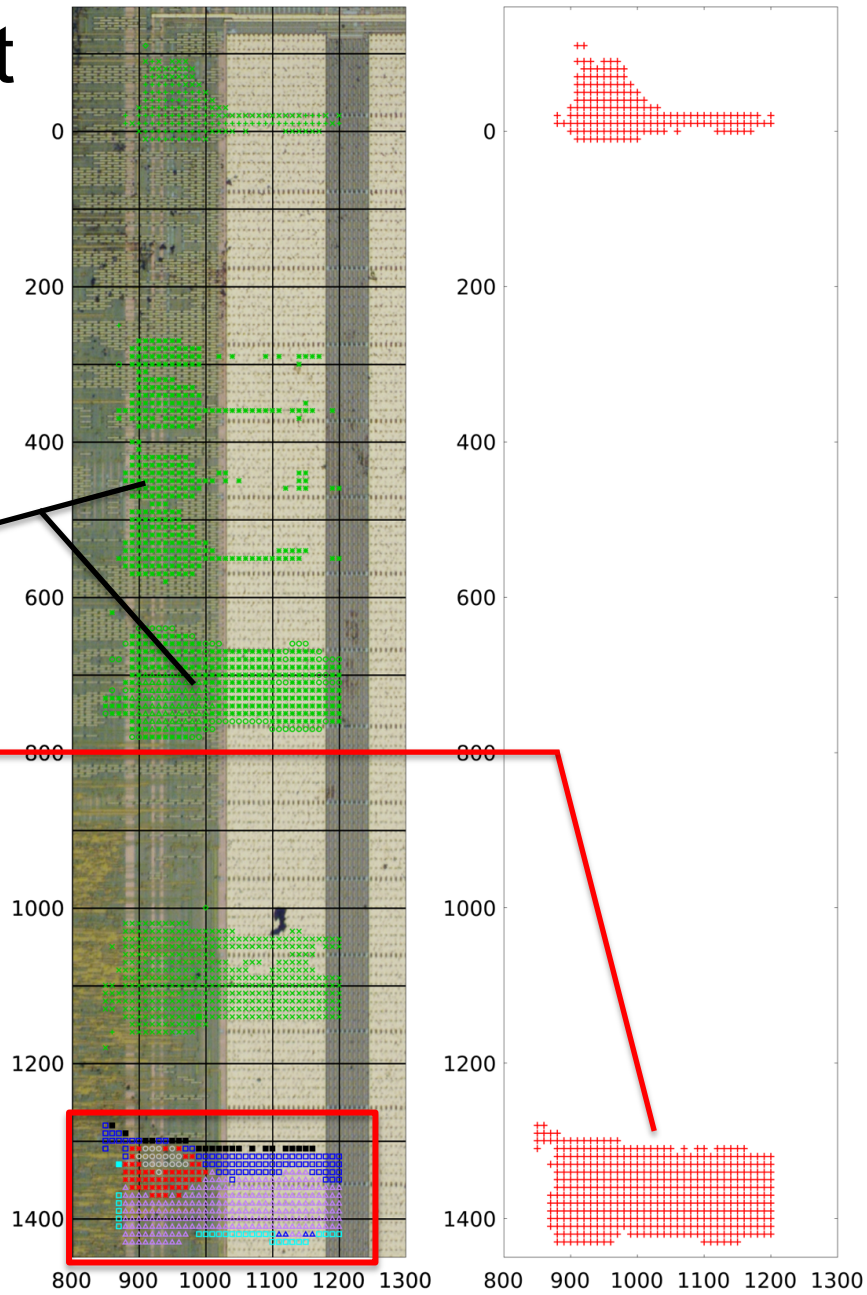
Laser settings: 200 ns (3x Tclk), 0.5 W
XY steps: **10 µm**
Synchronization: `ld r19, Z+`

S0	S1	S2	S3	S4	S5	S6	S7	S8	S9	SA
E0	E1	E2	E3	E4	E5	E6	E7	E8	E9	EA

S0: initialization value not found
→ not a skip

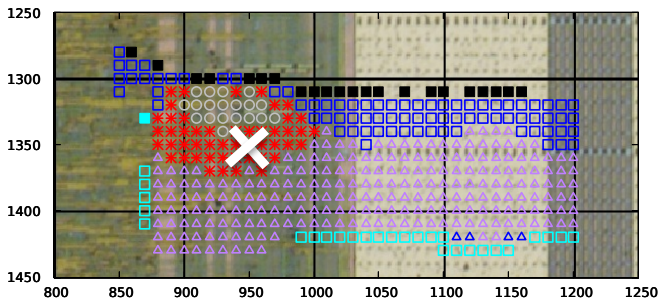
Trigger signals shortened
`ld: 2xTclk vs nop: 1xTclk`





❑ Chosen location

Laser settings: 75 ns, 0.4 W
Synchronization: `ld r19, Z+`



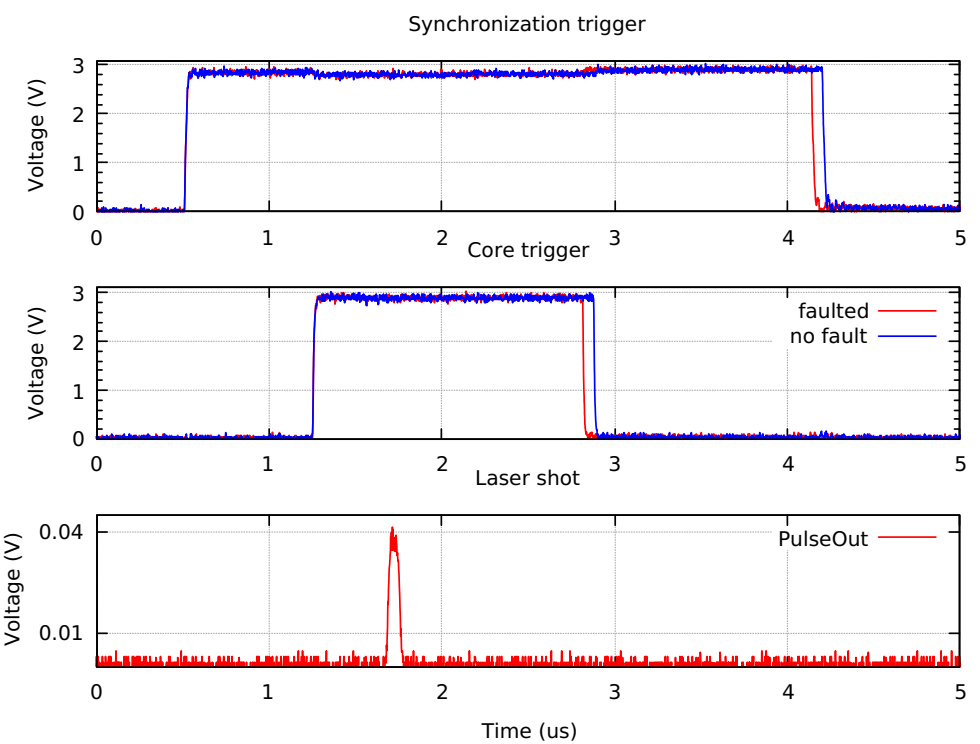

```
ld r16, Z+
ld r17, Z+
ld r18, Z+
nop
ld r20, Z+
...
ld r25, Z+
```

Laser-induced single nop

100% success rate

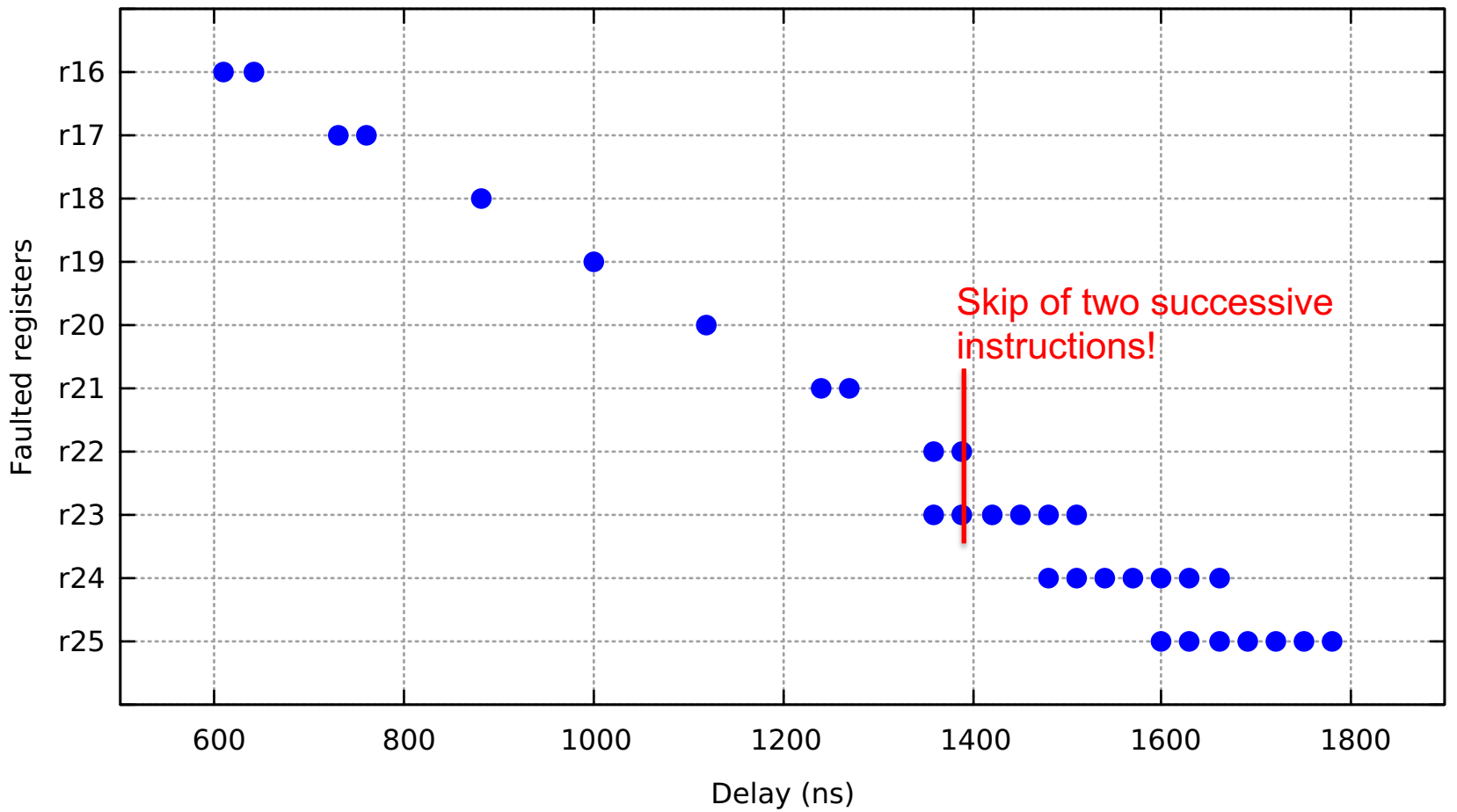
- Effect on triggers timing

`ld r19, Z+` 2xTclk → laser → nop 1xTclk



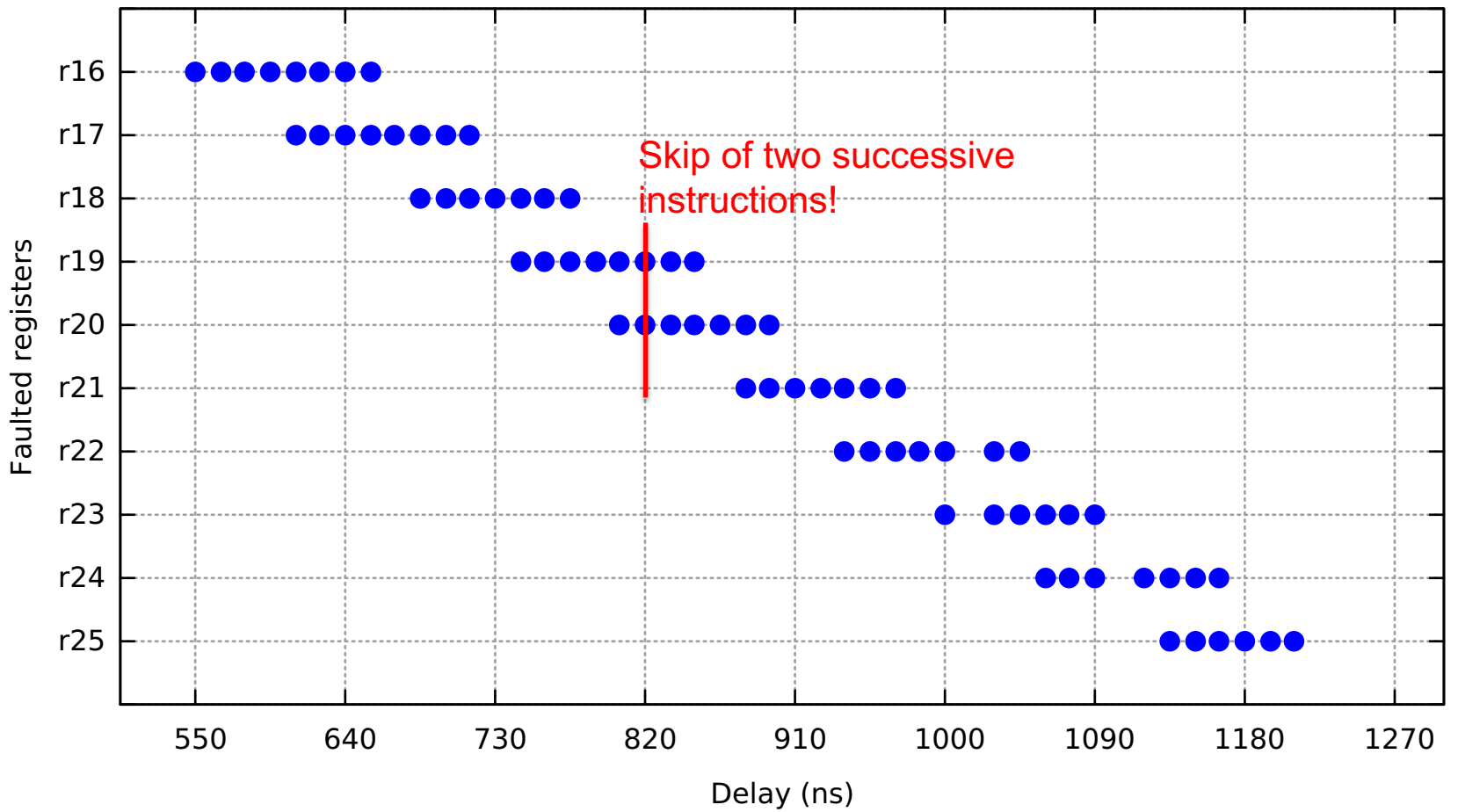
❑ Test of accuracy, ability to skip a given instruction

Laser settings: 75 ns, 0.4 W
100% success rate



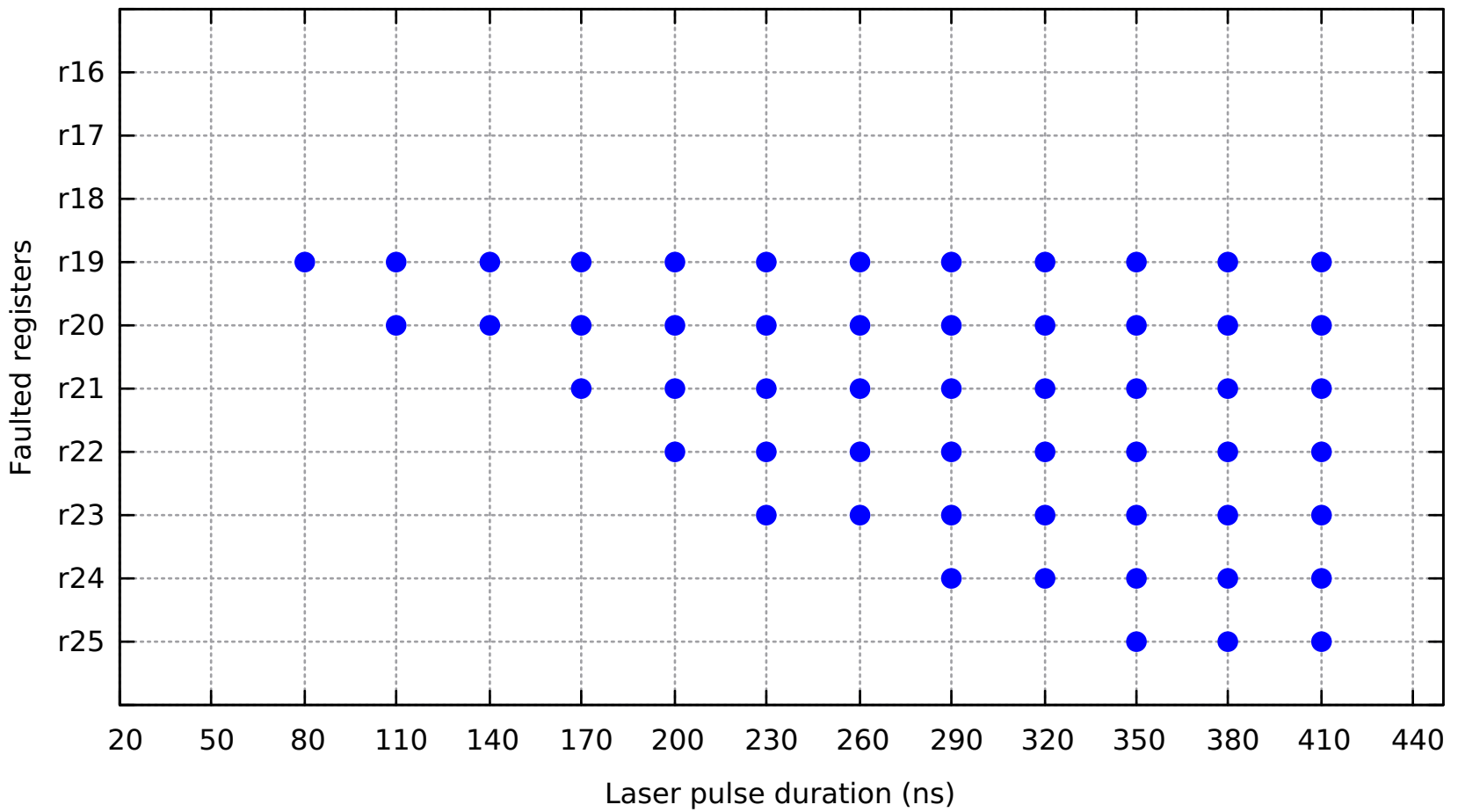
❑ Test of accuracy, ability to skip 2 instructions

Laser settings: **125 ns (2xTclk)**, 0.4 W
100% success rate



❑ Laser-induced arbitrary number of instr. skips

Laser settings: **80 ns to 410 ns**, 0.4 W
100% success rate



❑ Laser-induced arbitrary number of instr. skips

Laser settings: **50 ns to 20,400 ns**, 0.5 W

Laser pulse duration (ns)	1,000	2,000	5,000	10,000	20,400
Number of instr. Skips	17	33	82	143	300

❑ PIN bypass with several laser pulses

- PIN verification code*

`a1[i]` reference PIN
`a2[i]` user PIN

PIN arrays compare loop

```
BOOL diff = BOOL_FALSE
…
For (i=0; I < PIN_SIZE; i++) {
   if (a1[i] != a2[i]) {
      diff = BOOL_TRUE
   }
}
```

```
User PIN is incorrect
```


*From the ANR SERTIF project – http://sertif-projet.forge.imag.fr

❑ PIN bypass with several laser pulses

- Attack scenario

```
BOOL diff = BOOL_FALSE
…
For (i=0; I < PIN_SIZE; i++) {
   if (a1[i] != a2[i]) {
      diff = BOOL_TRUE
   }
}
```

User PIN is incorrect

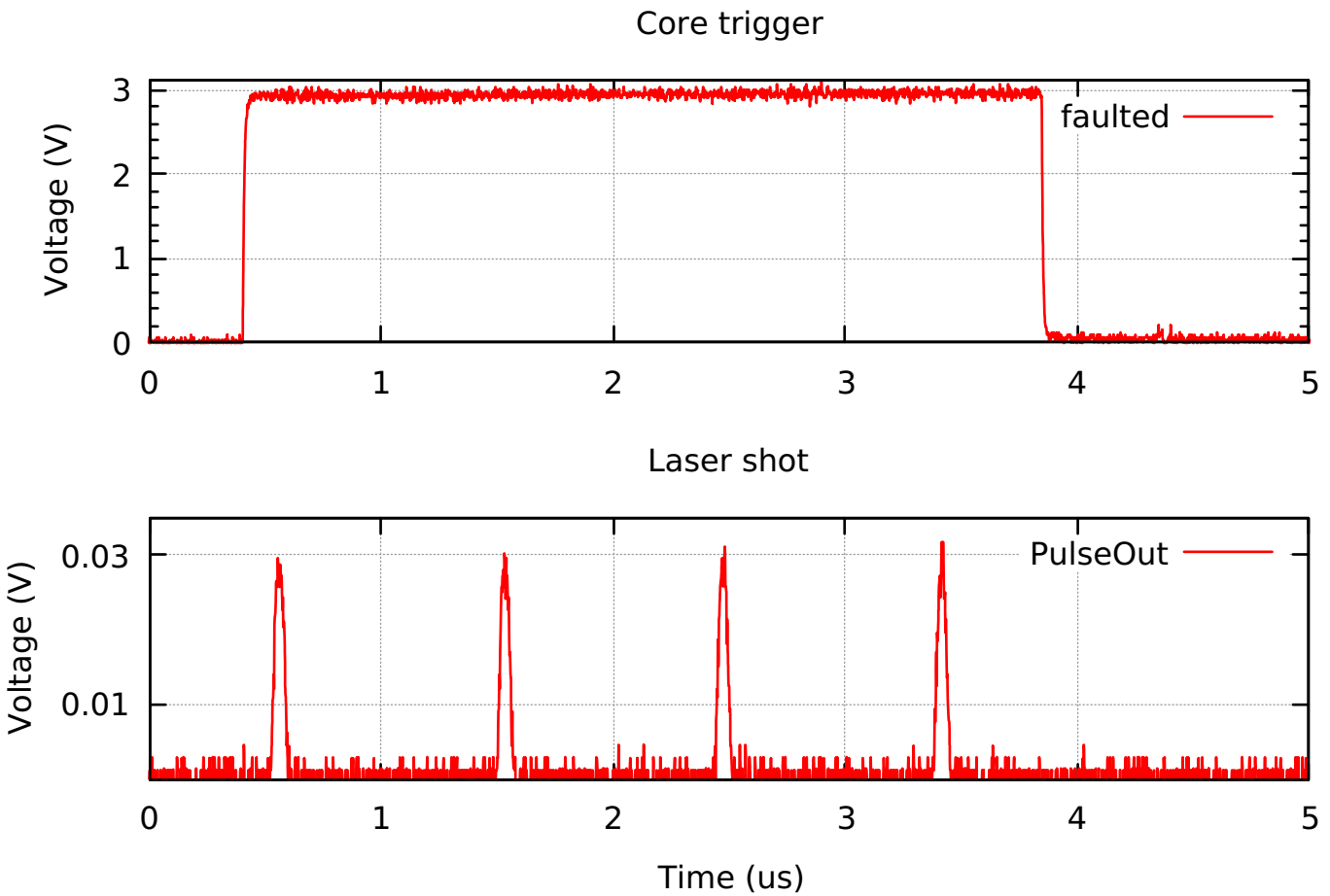
⇩ 4 separate laser pulses

```
BOOL diff = BOOL_FALSE
…
For (i=0; I < PIN_SIZE; i++) {
   if (a1[i] != a2[i]) {
      nop
   }
}
```

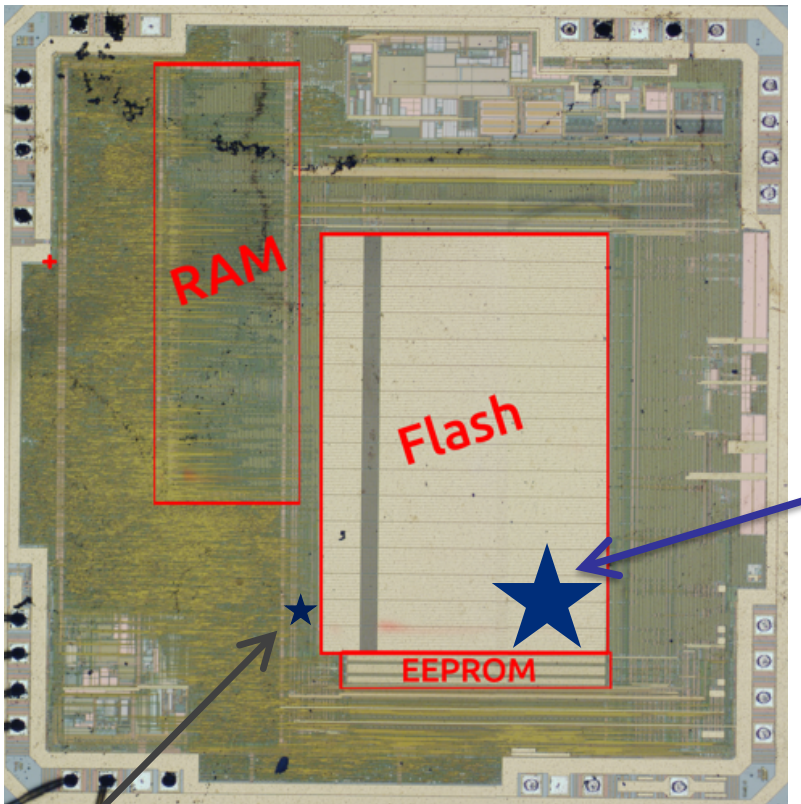
❑ PIN bypass with several laser pulses

Laser settings: 4x 60 ns, 0.5 W

Successful PIN bypass with 4 successive laser pulses (time between pulses: 875 ns)



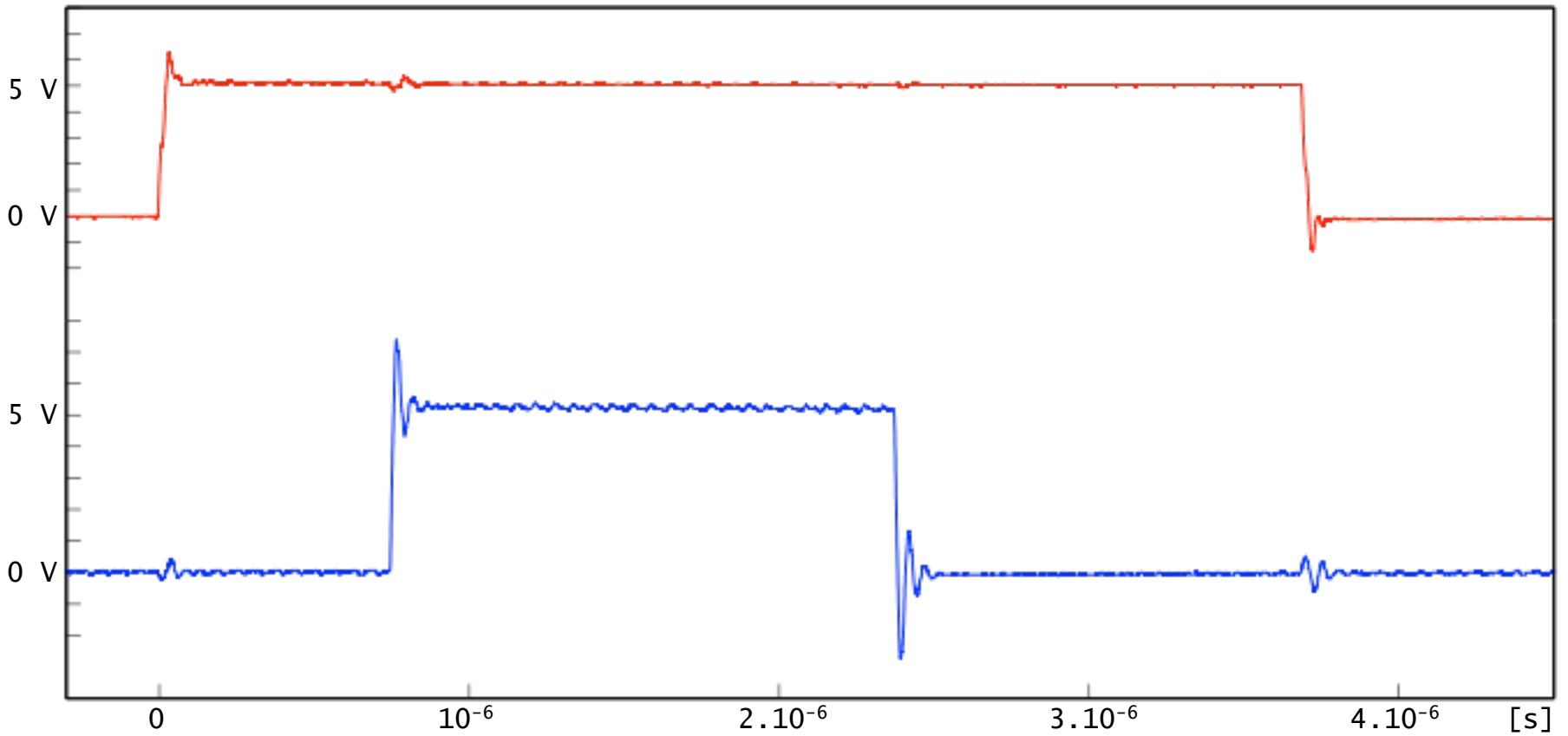
- EM-induced instruction skip
  - EM injection probe location



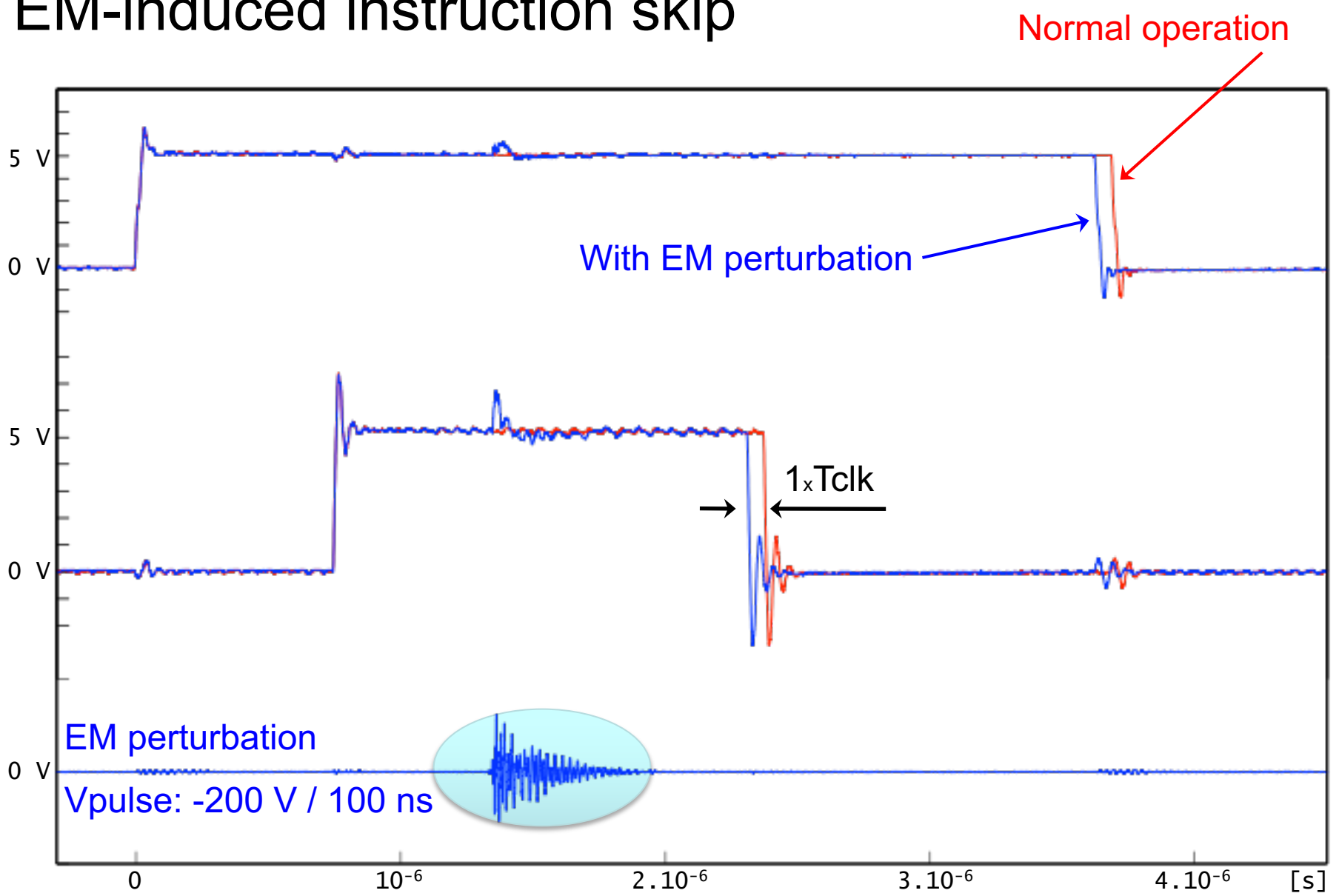
EM sensitive area
(used for reported exp. results)

Laser sensitive area

❑ EM-induced instruction skip



❑ EM-induced instruction skip



- EM-induced instruction skip (Vpulse: - 200 V / 100 ns)

1. Registers readback:

Register	16	17	18	19	20	21	22	23	24	25
Expected	0x39	0x38	0x37	0x36	0x35	0x34	0x33	0x32	0x31	0x30
Read	0x39	0x38	0x37	0x55	0x36	0x35	0x34	0x33	0x32	0x31

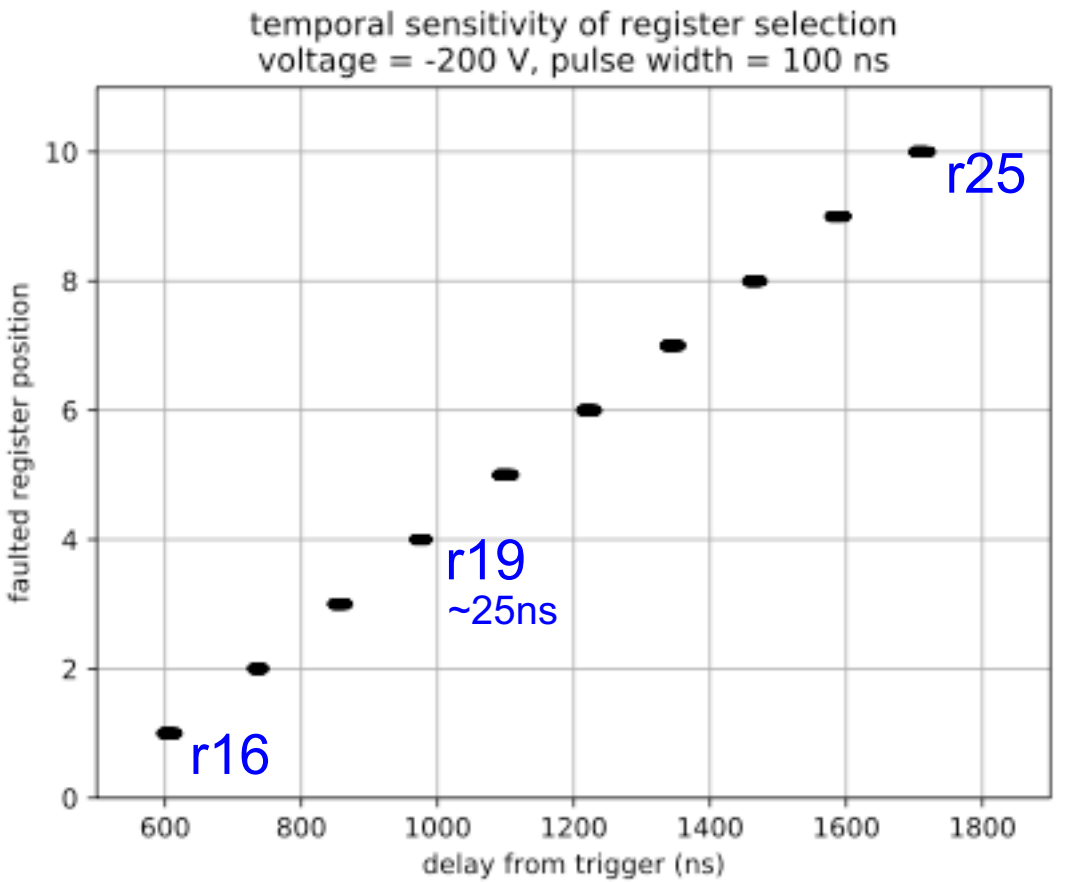
2. Decrement of trigger duration: 1$_x$Tclk

```
...                          ...
ld r18, Z+        EM         ld r18, Z+
ld r19, Z+        ⇨          nop
ld r20, Z+                   ld r20, Z+
...                          ...
```

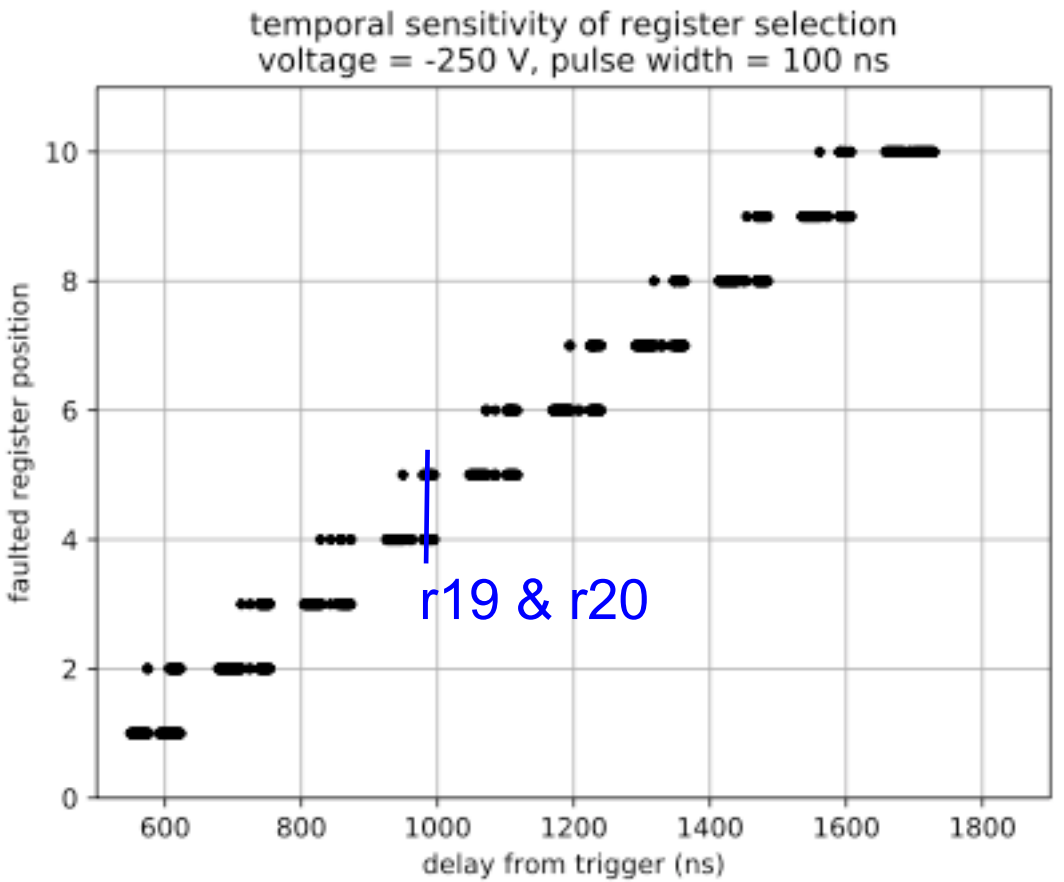
Instructions: ld → 2$_x$Tclk while nop → 1$_x$Tclk

Instruction skip: due to a nop **(not to an increment of the PC)**

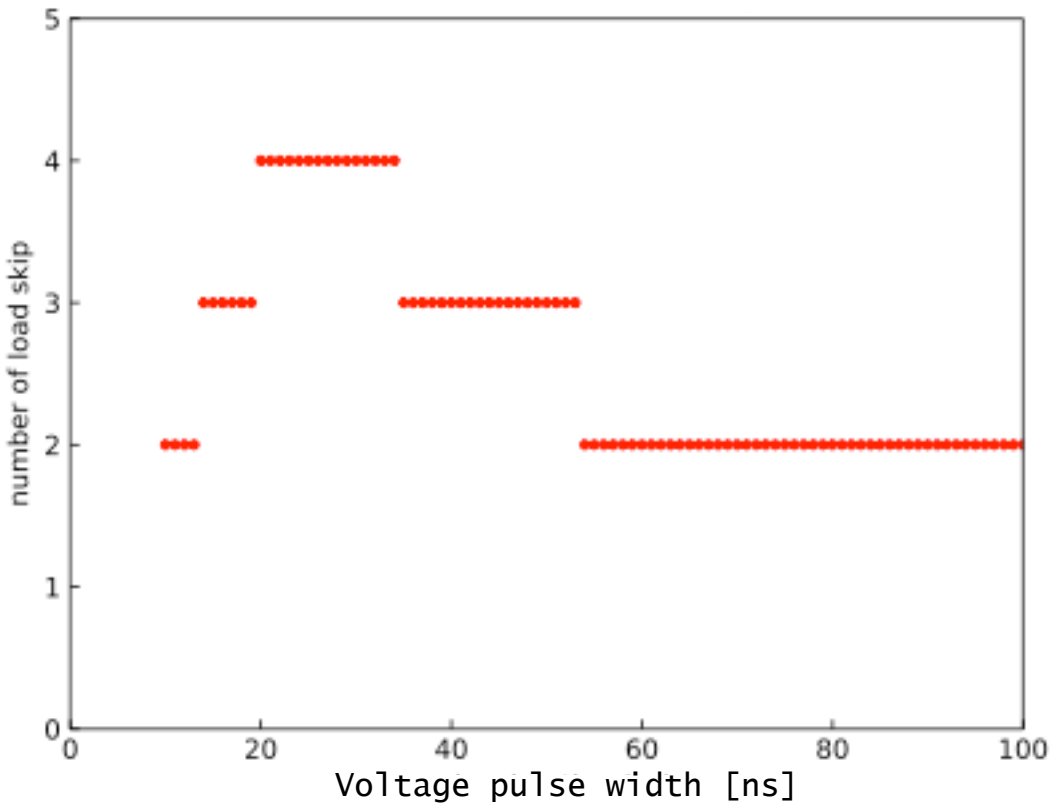
- EM-induced instruction skip (Vpulse: - 200 V / 100 ns)
  - Timing accuracy, single skip



- EM-induced instruction skip (Vpulse: **- 250 V** / 100 ns)
  - Double skip, obtained with an increased stress

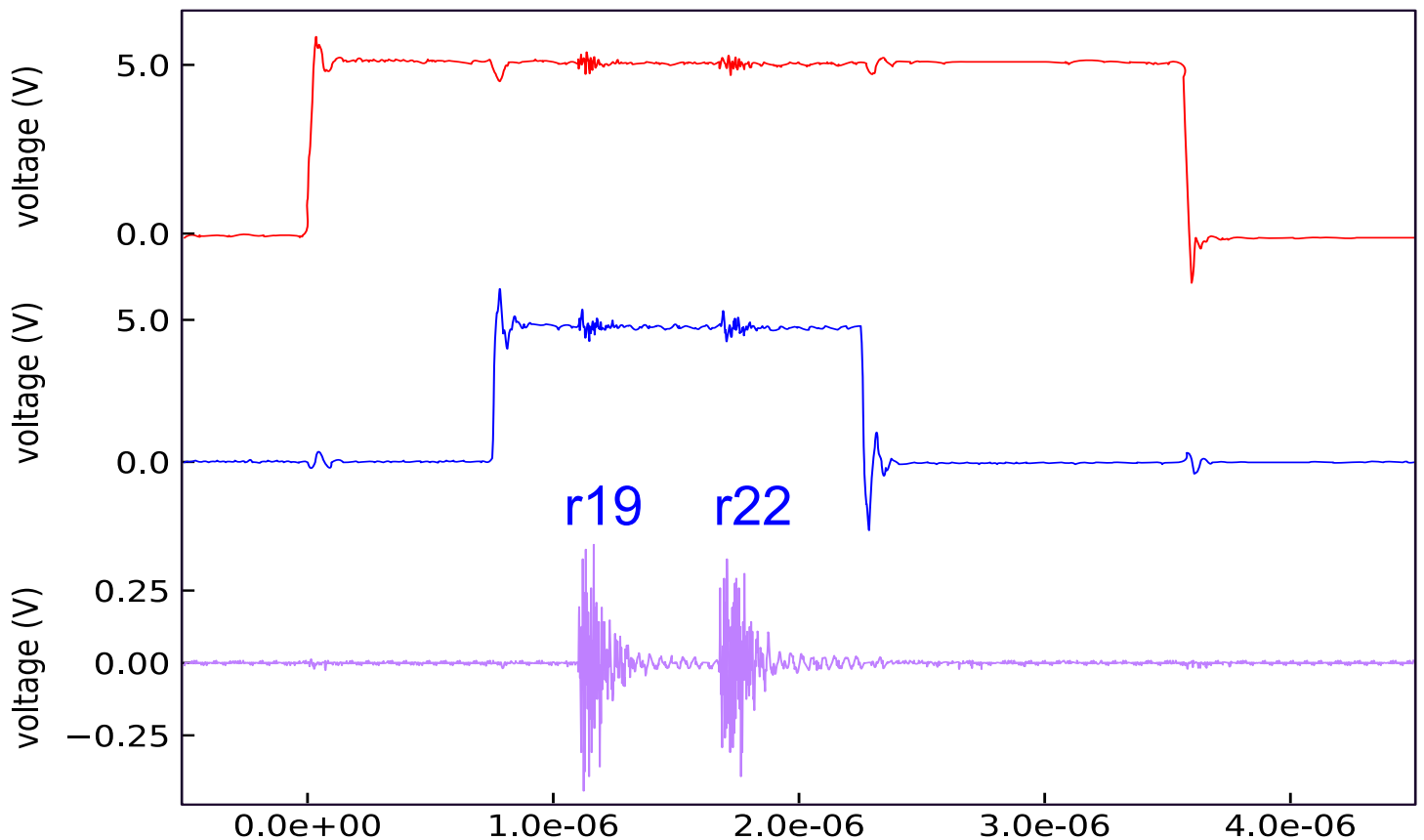


- EM-induced instruction skips (Vpulse: - 400 V)
  - Effect of modifying the voltage pulse width



→ Up to 6 consecutive instruction skips were obtained (other settings)

- EM-induced instruction skips (Vpulse: + 150 V, 10 ns, Δ = 570 ns)
  - Skipping none-consecutive instructions
    - Double pulse voltage generator: min. Δ = 150 ns (! usual repetition rate : 20 kHz)



III. Experimental results - Application to a verify PIN function

- Target's microcode: verify PIN function (x4: 0–9)

```
#define BOOL_TRUE 0xAA   \\10101010
#define BOOL_FALSE 0x55  \\01010101
```

```
void VerifyPIN()
{
 g_authenticated = BOOL_FALSE;

 if(g_ptc > 0) {
  if(byteArrayCompare(g_userPin, g_cardPin) == BOOL_TRUE) {
  g_ptc = 3;
  g_authenticated = BOOL_TRUE; // Authentication();
  }
 else {
   g_ptc--;
  }
 }
}
```

III. Experimental results - Application to a verify PIN function

- Target's microcode: verify PIN function (x4: 0–9)

```
#define BOOL_TRUE 0xAA   \\10101010
#define BOOL_FALSE 0x55  \\01010101
```

```
void VerifyPIN()
{
 g_authenticated = BOOL_FALSE;

 if(g_ptc > 0) {
  if(byteArrayCompare(g_userPin, g_cardPin) == BOOL_TRUE) {
  g_ptc = 3;
  g_authenticated = BOOL_TRUE; // Authentication();
  }
 else {
   g_ptc--;
  }
 }
}
```

g_authenticated
global var.
FALSE -> user
TRUE -> admin.

III. Experimental results - Application to a verify PIN function

- Target's microcode: verify PIN function (x4: 0–9)

```
#define BOOL_TRUE 0xAA   \\10101010
#define BOOL_FALSE 0x55  \\01010101
```

```
void VerifyPIN()
{
 g_authenticated = BOOL_FALSE;

 if(g_ptc > 0) {
  if(byteArrayCompare(g_userPin, g_cardPin) == BOOL_TRUE) {
  g_ptc = 3;
  g_authenticated = BOOL_TRUE; // Authentication();
  }
 else {
  g_ptc--;
  }
 }
}
```

g_ptc
pin try counter
max. = 3

III. Experimental results - Application to a verify PIN function

- Target's microcode: verify PIN function (x4: 0–9)

```
#define BOOL_TRUE 0xAA   \\10101010
#define BOOL_FALSE 0x55  \\01010101
```

```
void VerifyPIN()
{
 g_authenticated = BOOL_FALSE;

 if(g_ptc > 0) {
  if(byteArrayCompare(g_userPin, g_cardPin) == BOOL_TRUE) {
  g_ptc = 3;
  g_authenticated = BOOL_TRUE; // Authentication();
  }
 else {
   g_ptc--;
  }
 }
}
```

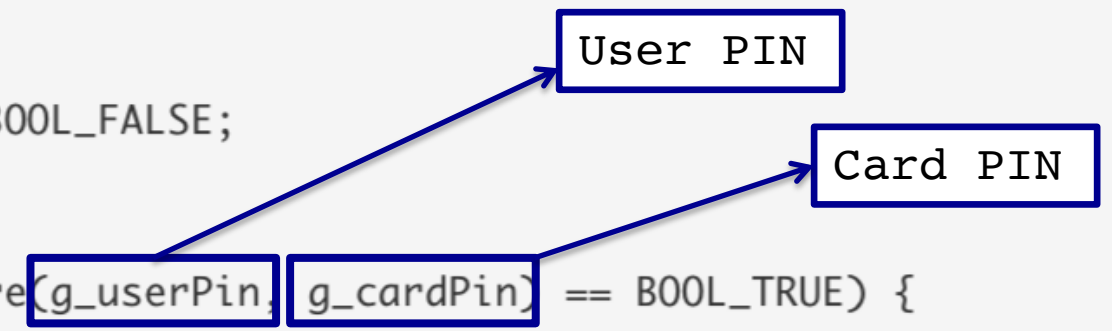
III. Experimental results - Application to a verify PIN function

- Target's microcode: verify PIN function (x4: 0–9)

```
#define BOOL_TRUE 0xAA   \\10101010
#define BOOL_FALSE 0x55  \\01010101
```

```
void VerifyPIN()
{
 g_authenticated = BOOL_FALSE;

 if(g_ptc > 0) {
  if(byteArrayCompare(g_userPin, g_cardPin) == BOOL_TRUE) {
  g_ptc = 3;
  g_authenticated = BOOL_TRUE; // Authentication();
  }
 else {
   g_ptc--;
  }
 }
}
```



III. Experimental results - Application to a verify PIN function

- Attack path: test of the remaining number of tries
  - Force strings comparison → PIN exhaustive search (brute force attack)
  - EM-induced instruction skip

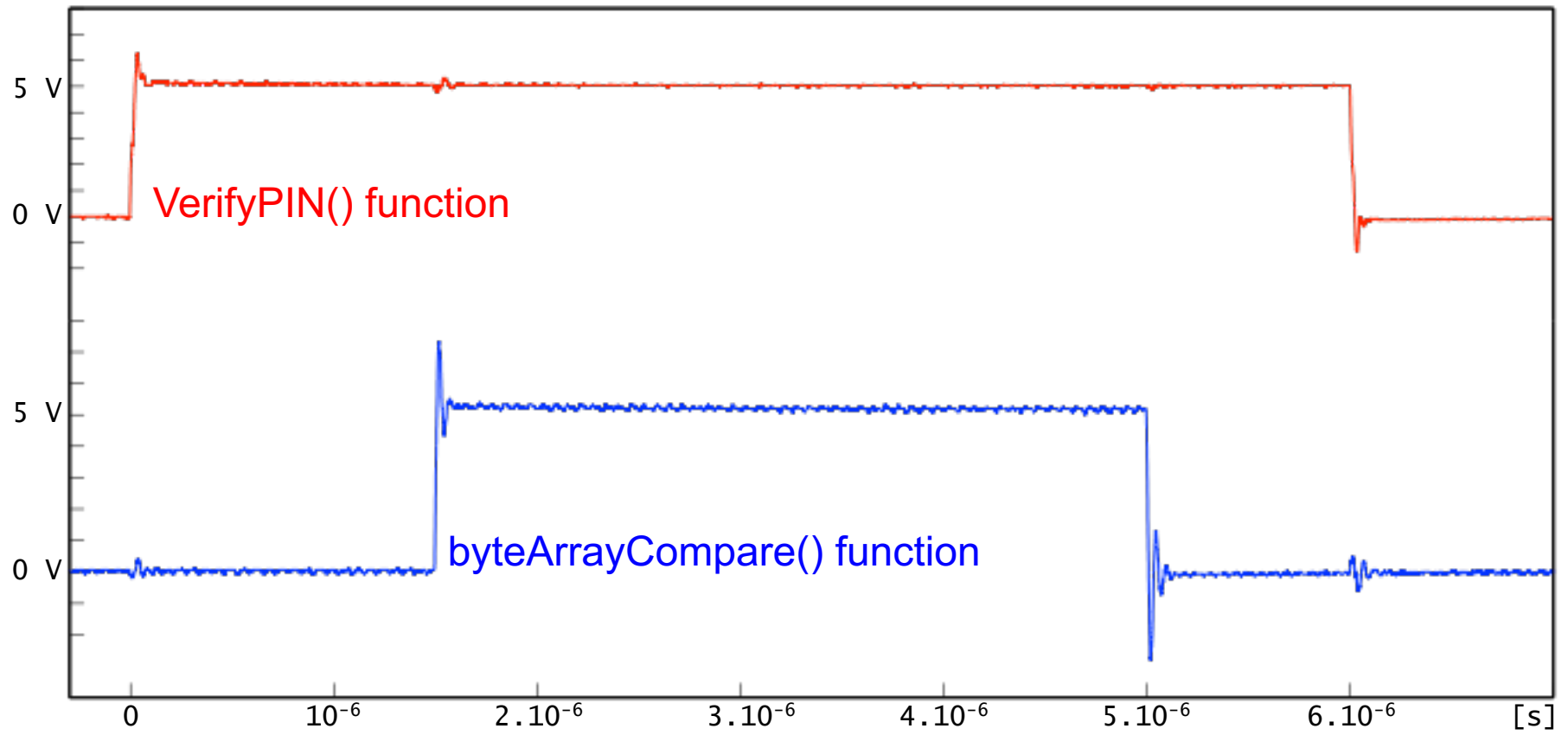
```
void VerifyPIN()
{
 g_authenticated = BOOL_FALSE;

 if(g_ptc > 0) {
  if(byteArrayCompare(g_userPin, g_cardPin) == BOOL_TRUE) {
  g_ptc = 3;
  g_authenticated = BOOL_TRUE; // Authentication();
  }
 else {
   g_ptc--;
  }
 }
}
```

```
g_ptc
pin try counter
max. = 3
```

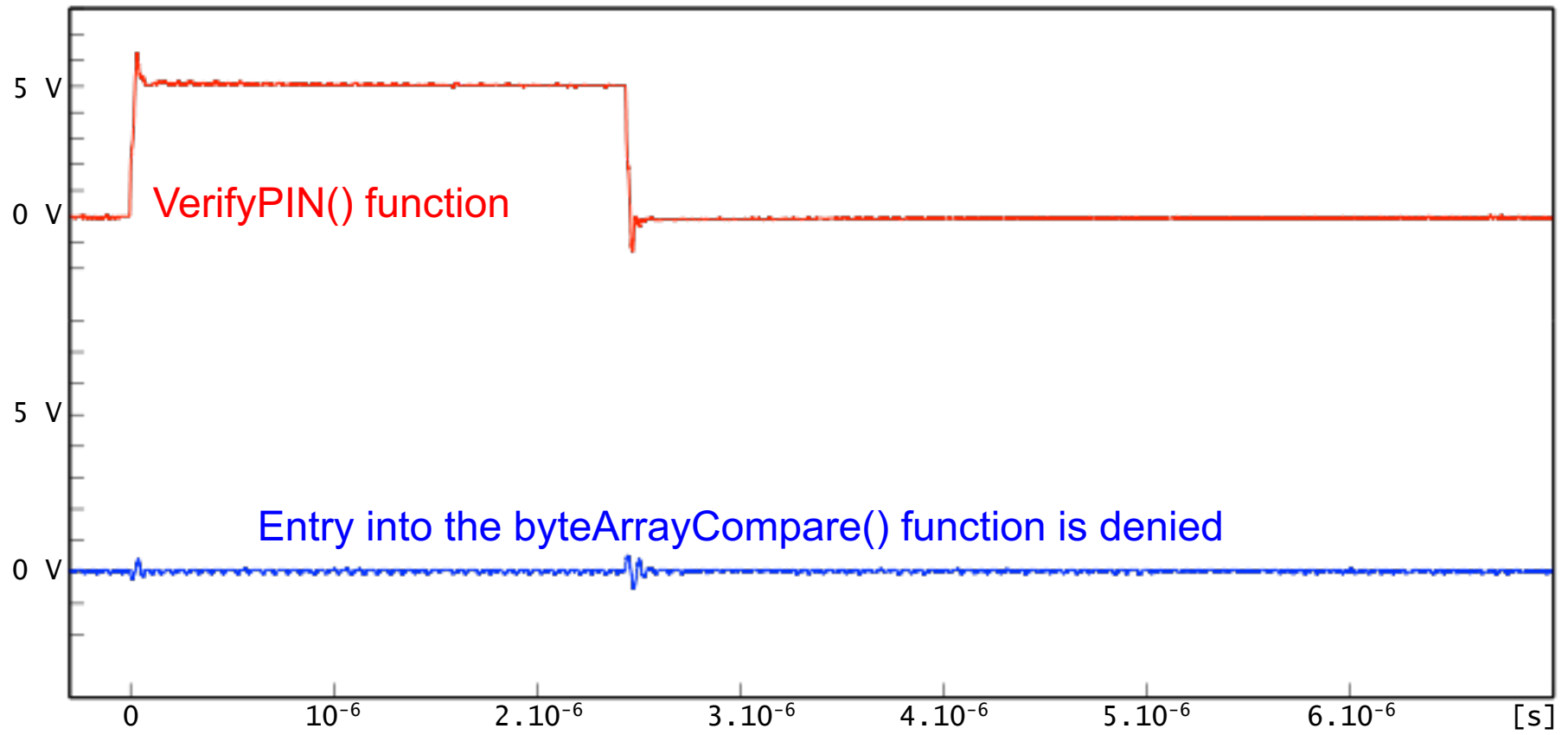
III. Experimental results - Application to a verify PIN function

- Trigger waveforms: g_ptc > 0



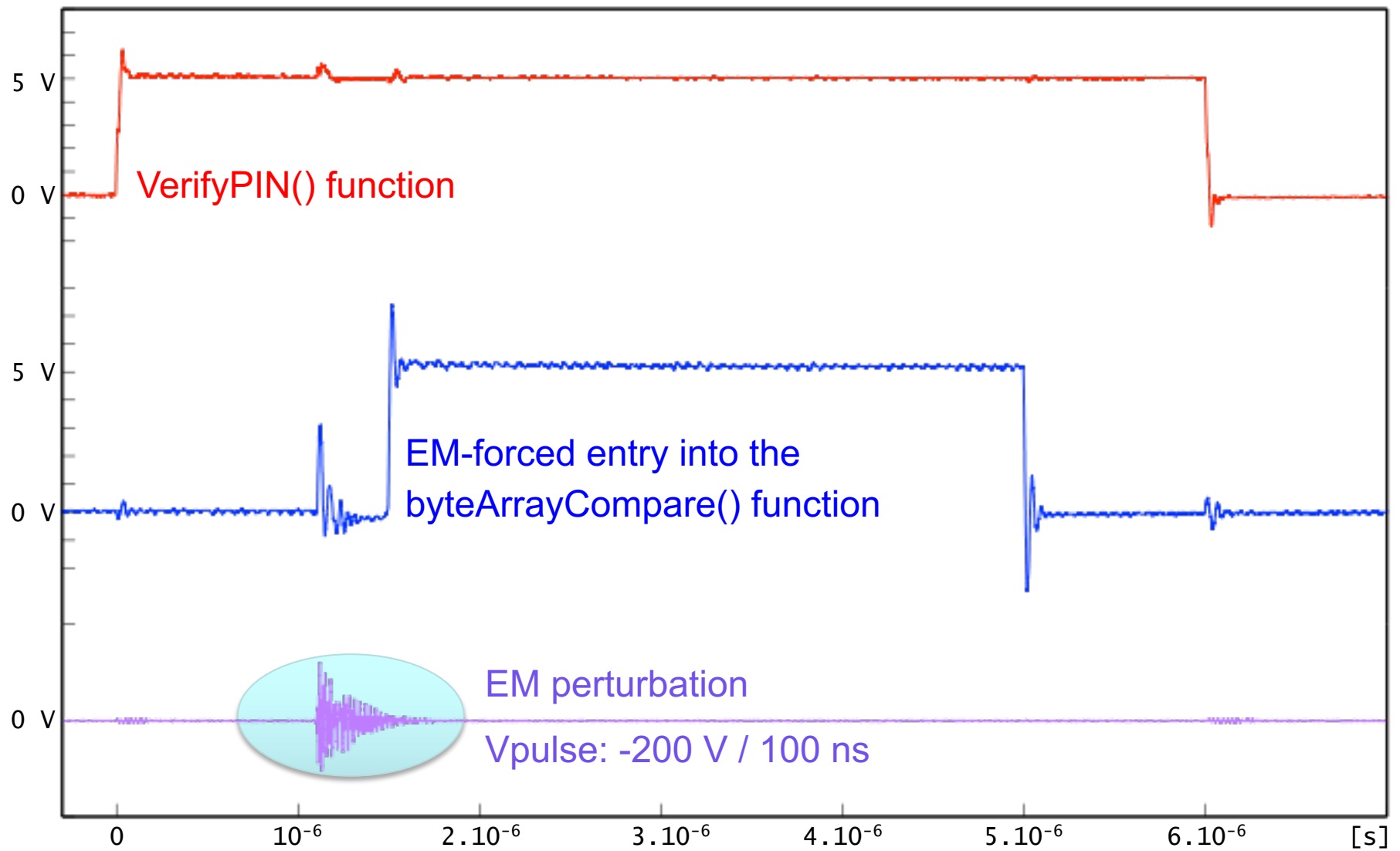
III. Experimental results - Application to a verify PIN function

- Trigger waveforms: g_ptc = 0



III. Experimental results - Application to a verify PIN function

- g_ptc = 0 + EM attack → exhaustive search (success rate 100%)



Outline

I. Laser-induced instruction skip fault model

II. Experimental settings

III. Experimental results

IV. Discussion

V. Conclusion

❑ Laser/EM-induced instruction skip fault model

▪ A powerful fault model

Experimental assessment on an 8-bit microcontroller

- ability to skip a (chosen) single instruction with 100% success rate
- ability to skip an arbitrary number of successive instructions (laser)

  (only 6 successive skips for EM injection, on exp. Basis)
- ability to skip different sections of a microcode

Requirement: synchronization with the target

▪ Denomination of the fault model

Instructions are not skipped, rather turned into a nop

Also instruction modification (turning an instruction into another one irrelevant to the actual part of the program).

❑ Synchronization

Easy when using trigger signal (white box approach)

Black box?

- Use of a smart trigger (real time recognition of a side channel signal, [WOU11])

[WOU11] van Woudenberg et al., Practical optical fault injection on secure microcontrollers, FDTC 2011

❑ Synchronization

Easy when using trigger signal (white box approach)

Black box?

- Use of a smart trigger (real time recognition of a side channel signal, [WOU11])

❑ Generalization

Generalization on a modern microcontroller: still to be proven

However:

- [TRI10] reports similar single skip near the Flash of a 32-bit microcontroller
- [COL19] highlights a mechanism that shall allow multi-skips injection
- [DUT18] obtained single bit faults on a CMOS 28nm target

[TRI10] Trichina et al., Multi fault laser attacks on protected CRT-RSA, FDTC 2010
[COL19] Colombier et al., Laser-induced single-bit faults in flash memory: Instructions corruption on a 32-bit microcontroller HOST2019
[DUT18] Dutertre et al., Laser fault injection at the CMOS 28 nm technology node: an analysis of the fault model, FDTC 2018

I. Laser-induced instruction skip fault model

II. Experimental settings

III. Experimental results

IV. Discussion

V. Conclusion

- Experimental assessment of extended laser/EM-induced instruction skip fault models

Laser:

Ability to erase chosen parts of a microcontroller firmware at runtime

EM:

Ability to erase a chosen instruction (or small groups of instr.) in a microcontroller program at runtime

- To be considered when designing countermeasures

Software CMs may be also skipped if not designed carefully

dutertre@emse.fr

To go further:

Experimental analysis of the laser-induced in- struction skip fault model
J.-M. Dutertre, T. Riom, O. Potin, and J.-B. Rigaud, Nordsec 2019

Experimental analysis of the electromagnetic instruction skip fault model
A. Menu, J.-M. Dutertre, O. Potin, J.-B. Rigaud, and J.-L. Danger, DTIS 2020

This research has been partially supported by the European Commission under H2020 SPARTA (Grant Agreement 830892)

MINES Saint-Étienne
Une école de l'IMT

Institut Mines-Télécom

Département Systèmes et Architectures Sécurisées
Mines Saint-Etienne,
Centre de Microélectronique de Provence
13541 Gardanne FRANCE